**Network protocols**

1. Layers of OSI Model (7)
   1. **Application layer**
   2. Presentation Layer
   3. Session Layer
   4. **Transport Layer**
   5. **Network Layer**
   6. Data Link Layer
   7. Physical Layer

**Application Layer protocols**

1. Application Layer protocols
   1. Client-Server Protocols
      1. **HTTP/HTTPS**
      2. FTP
      3. SMTP
      4. **WebSockets**
   2. Peer 2 Peer protocols
      1. **WebRTC**
2. Client-Server protocols (except Web Socket)
   1. **Unidirection communication**
      1. Client (Web browser) initiates the connection – Sends request
      2. Server (Web server) – Response
   2. **HTTP**
      1. Connection oriented (1 connection)
      2. Used to access web pages, jump from one page to another via hyperlinks
   3. FTP
      1. 2 connections maintained (Control and Data connection)
      2. Data connection can be created and let go, but Control connection persists
      3. Useless – Data is not encrypted in Data connection
   4. SMTP (in combination with IMAP)
      1. For sending / receiving emails
      2. SMTP ; Send email
      3. IMAP : receive email (POP3 is old and not used anymore since it used to delete email after downloading on one device)
      4. User Agent (sender) -> send mail to MTA client -> forward to MTA Server -> deliver to User Agent (receiver)
3. Web Socket
   1. **Bidirectional communication**
      1. Client can talk to server, server can talk to client
      2. Different from P2P (where even client can talk to client)
   2. **Usecase : Messaging App** (since client doesn’t have to keep asking if new message came for it!)
4. Peer 2 Peer protocol - **WebRTC**
   1. All can talk to all, Client-Server, Server-Client, Client-Client
   2. Fast
   3. **Usecase : Video calling, Livestreaming**

**Transport/Network Layer protocols**

1. Transport/Network Layer protocols
   1. **TCP/IP**
   2. **UDP/IP**
2. TCP/IP
   1. Maintain a virtual connection through which data flows
   2. Break into packets -> add sequence no. to packets -> send to destination -> **ensure order** using sequence no. -> **Retry/Resend packets** for which we didn’t get Acknowledgement from receiver
   3. Characteristics : Packets, connection maintain, ordering, Acknowledgement
3. UDP/IP
   1. Divide into Datagrams (here we don’t call them Packets!) -> send over multiple parallel virtual connections
   2. No ordering, lost packets, no connection maintain, no Acknowledgement – **FAST**
   3. **Usecase : Video Calling, LiveStreaming** (where I don’t want to see lost packets)
   4. **WebRTC**(P2P protocol) **uses UDP**